인공지능과 라즈베리파이 기술을 이용한 전동 킥보드 수동/자동 SOS 신고 서비스 개발

|  |
| --- |
| **설계서** |

2023.11.30

가천대학교 컴퓨터공학과

팀장 ◼︎ 류성열 202035147

팀원 ◼︎ 손현철 201735984

팀원 ◼︎ 안지훈 201935288

팀원 ◼︎ 정찬진 201736036

문서 정보

|  |
| --- |
| 본 문서는 인공지능과 라즈베리파이 기술을 이용한 전동 킥보드 수동/자동 SOS 신고 서비스 개발을 위한 요구규격 및 설계서입니다. |

|  |  |
| --- | --- |
| 버전 | 0.2 |
| 작성일 | 2023.11.29 |
| 상태 | ◻︎완료 ◼진행중 ◻︎초안 |

|  |  |  |  |
| --- | --- | --- | --- |
| **버전** | **변경한 사람** | **변경한 날짜** | **버전업 변경(또는 추가)내용** |
| 0.2 | 류성열 | 2023-11-29 |  |
| 손현철 |
| 안지훈 |
| 정찬진 |
|  | 류성열 | 2023-11-30 |  |
| 손현철 |
| 안지훈 |
| 정찬진 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

목 차

**2.4 요구사항 기능 정의 (사용자)**

**○** 소프트웨어 요구사항 기능 정의

표 2.4 사용자 관련 기능 정의표

|  |  |  |  |
| --- | --- | --- | --- |
| 번호 | 주요기능명 | 상세기능명 | 비고 |
| 1) | 사용자 인증 기능 | 회원가입 |  |
| 2) | 로그인 |  |
| 3) | 로그아웃 |  |
| 4) | 교통사고 신고 기능 | 자동신고 온/오프 |  |
| 5) | 번호 관리 기능 | 번호 등록 |  |
| 6) | 번호 삭제 |  |

1) 회원가입

- 사용자가 앱을 이용하기 위해 사용자 정보를 등록한다.

- 사용자는 앱의 로그인 페이지에서 회원가입 버튼을 통해 회원가입을 수행할 수 있다.

- 회원가입은 이름, 아이디, 비밀번호, 전화번호의 개인정보를 입력 받는다.

- 사용자로부터 받은 정보는 데이터베이스에 저장된다.

- 회원가입 후 로그인을 통해 사용자는 제공되는 모든 서비스를 이용할 수 있다.

2) 로그인

- 사용자는 앱을 이용하기 위해 로그인을 수행해야 한다.

- 사용자는 앱의 로그인 페이지에서 로그인 버튼을 통해 로그인을 할 수 있다.

- 로그인은 사용자 아이디, 사용자 비밀번호로 이루어져 있고, 사용자로부터 입력 받아서 데이터베이스에 저장된 값과 비교한다.

- 비교 결과에 따라 로그인 성공 혹은 실패 화면을 보여준다.

- 사용자는 로그인으로 사용자 인증을 마친 후에 제공되는 모든 서비스를 이용할 수 있다.

3) 로그아웃

- 사용자는 로그아웃 버튼, 혹은 앱의 종료를 통해 로그아웃을 수행할 수 있다.

- 사용자의 로그인 정보를 세션에서 종료 시킨다.

- 사용자가 로그아웃을 수행한 경우, 다시 로그인을 해야 모든 서비스를 이용할 수 있다.

4) 자동신고 온/오프

- 사용자는 자동신고 기능을 활성화하거나 비활성화할 수 있다.

- 사용자가 자동신고를 끄면 교통사고가 감지되어도 신고를 하지 않는다.

- 사용자가 자동신고를 키면 교통사고가 감지되었을 경우 자동으로 신고한다.

5) 번호 등록

- 사용자는 신고 시에 저장된 번호로 사고 소식을 알릴 수 있다.

- 번호 등록 페이지에서 이름, 번호의 정보를 입력받고, 입력받은 정보는 데이터베이스에 저장된다.

- 번호 등록 완료 후 교통사고가 감지되었을 경우 저장된 번호로 사고 소식을 전송한다.

6) 번호 삭제

- 사용자는 저장된 번호를 삭제할 수 있다.

- 사용자가 번호 삭제를 수행한 경우 저장된 정보는 데이터베이스에서 삭제된다.

**2.5 요구사항 기능 정의 (관리자)**

**○ 소프트웨어 요구사항 기능 정의**

표 2.5 관리자 관련 기능 정의표

|  |  |  |  |
| --- | --- | --- | --- |
| 번호 | 주요기능명 | 상세기능명 | 비고 |
| 1) | 관리자 인증 기능 | 로그인 |  |
| 2) | 로그아웃 |  |
| 3) | 회원 관리 기능 | 회원 정보 조회 |  |
| 4) | 센싱 데이터 관리 기능 | 센싱 데이터 조회 |  |

1) 로그인

- 관리자는 서비스를 이용하기 위해 로그인을 수행해야한다.

- 로그인은 관리자 아이디, 관리자 비밀번호로 이루어져 있고 관리자로부터 입력받아 데이터베이스에 저장된 값과 비교한다.

- 비교 결과에 따라 로그인 성공 혹은 실패 화면을 보여준다.

- 관리자는 로그인으로 관리자 인증을 마친 후에 제공되는 모든 서비스를 사용할 수 있다.

2) 로그아웃

- 관리자는 로그아웃 버튼, 혹은 종료를 통해 로그아웃을 수행할 수 있다.

- 관리자의 로그인 정보를 세션에서 종료시킨다.

- 관리자가 로그아웃을 수행한 경우 다시 로그인을 해야 서비스를 이용할 수 있다.

3) 회원 정보 조회

- 관리자는 모든 회원의 정보를 데이터베이스에서 조회할 수 있다.

- 관리자 페이지에 회원의 정보가 테이블 형태로 출력된다.

4) 센싱 데이터 조회

- 관리자는 모든 센싱 데이터를 데이터베이스에서 조회할 수 있다.

- 관리자 페이지에 센싱 데이터가 테이블 형태로 출력된다.

**2.6 요구사항 기능 정의 (AI Server)**

**○ 소프트웨어 요구사항 기능 정의**

표 2.5 AI Server 관련 기능 정의표

|  |  |  |  |
| --- | --- | --- | --- |
| 번호 | 주요기능명 | 상세기능명 | 비고 |
| 1) | 교통사고 분류 기능 | 데이터셋 생성 |  |
| 2) | 모델 구축 |  |
| 3) | 교통사고 분류 |  |

1) 데이터셋 생성

- 사용자에게 더 나은 교통사고 분류 기능을 제공하고, 모델 학습을 위해 품질이 좋은 데이터셋을 생성한다.

- 데이터셋은 4개의 특징(속도, 기울기, 진동, 거리)과 타깃값(0, 1) 쌍으로 이뤄져있고, 총 20,000개의 표본으로 설정한다.

- 데이터셋의 비율은 훈련 : 검증 : 테스트 = 70:20:20로 설정한다.

- 각 특징값은 랜덤값으로 생성하며, 그에 대한 타깃값은 임의로 설정한다.

2) 모델 구축

- 사용자에게 더 나은 교통사고 분류 기능을 제공하기 위해 여러 모델로 실험한다.

- 이를 통해 가장 성능이 좋은 모델을 선정하고, 테스트 세트로 검증을 실시한다.

- 검증 점수는 정확도 0.9(백분율) 이상을 목표로 하며, 미달 시 재학습한다.

3) 교통사고 분류

- 사용자는 교통사고 분류 기능을 제공받을 수 있다.

- IoT 기기에서 AI Server로 센싱 데이터를 보내면 모델은 교통사고를 판별한다.

- 교통사고로 판별될 경우 데이터베이스에 센싱 데이터를 저장한다.

**2.7 AI 모델 구성 (Python Library)**

**○ Pandas**

Pandas 라이브러리는 데이터 분석을 할 때 가장 많이 쓰이는 패키지다. 대부분의 데이터는 시계열(Series)이나 표(Table)의 형태로 나타낼 수 있는데 Pandas 패키지에서는 이러한 데이터 표 데이터를 다루기 위한 시리즈(Series) 클래스와 데이터 프레임(Data Frame) 클래스를 제공한다.



그림 2.7.1 Pandas Library

**○ NumPy**

NumPy 라이브러리는 2005년에 Travis Oliphan가 발표한 수치해석용 Python 패키지이다. 다차원의 행렬 자료구조인 ndarray를 지원하여 벡터와 행렬을 사용하는 선형대수 계산에 주로 사용한다. 내부적으로 BLAS 라이브러리와 LAPACK 라이브러리에 기반하고 있어서 C로 구현된 CPython에서만 사용할 수 있으며 Jython, IronPython, PyPy 등의 Python 구현에서는 사용할 수 없다. Numpy의 행렬 연산은 C로 구현된 내부 반복문을 사용하기 때문에 Python 반복문에 비해 속도가 빠르다. 행렬 인덱싱(Array indexing)을 사용한 질의(Query) 기능을 이용하여 짧고 간단한 코드로 복잡한 수식을 계산할 수 있다.

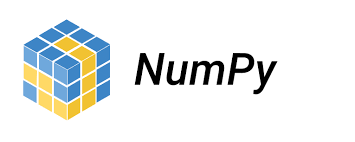


그림 2.7.2 NumPy Library

**○ Matplotlib**

Matplotlib 라이브러리는 Python에서 자료를 차트(Chart)나 플롯(Plot)으로 시각화(Visualization)하는 패키지이다. Matplotlib은 다음과 같은 정형화된 차트나 플롯 이외에도 저 수준 api를 사용한 다양한 시각화 기능을 제공한다.



그림 2.7.3 Matplotlib Library

**○ Tensorflow**

텐서플로우(Tensorflow)는 데이터 플로우 그래프(Data flow graph)를 사용하여 수치 연산을 하는 오픈소스 소프트웨어 라이브러리이다. 그래프의 노드(Node)는 수치 연사을 나타내고 엣지(Edge)는 노드 사이를 이동하는 다차원 데이터 배열(Tensor)를 나타낸다. 유연한 아키텍처로 구성되어 있어 코드 수정 없이 데스크탑, 서버 혹은 모바일 디바이스에서 CPU나 GPU를 사용하여 연산을 구동시킬 수 있다. 텐서플로우는 원래 머신러닝과 딥 뉴럴 네트워크 연구를 목적으로 구글의 인공지능 연구 조직인 구글 브레인 팀의 연구자와 엔지니어들에 의해 개발되었다.

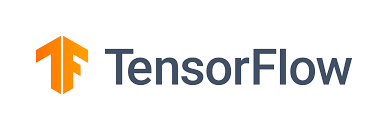


그림 2.7.4 Tensorflow Library

**○ Keras**

케라스(Keras)는 파이썬으로 작성된 오픈 소스 신경망 라이브러리이다. MXNet, Deeplearning4j, 텐서플로, Microsoft Cognitive Toolkit 또는 Theano 위에서 수행할 수 있다. 딥 신경망과의 빠른 실험을 가능케 하도록 설계되었으며 최소한의 모듈 방식의 확장 가능성에 초점을 둔다. ONEIROS 프로젝트의 연구적 노력의 일환으로 개발되었다.



그림 2.7.5 Keras Library

**○ Scikit-learn**

사이킷런(Scikit-learn)은 파이썬에서 사용할 수 있는 머신러닝 라이브러리로, 간단하고 효과적인 도구를 제공하여 데이터 분석과 머신러닝 모델 개발을 지원한다. 다양한 머신러닝 알고리즘, 데이터 전처리 기능, 모델 평가 도구 등을 포함하고 있어 데이터 과학 및 머신러닝 프로젝트에서 널리 사용된다. 파이썬 생태계에서 가장 인기 있는 머신러닝 라이브러리 중 하나로, 문서화가 잘 되어 있고 활발하게 개발되고 있다. 또한, 간단한 API 구조와 예제 코드를 통해 사용자가 머신러닝 모델을 쉽게 구축할 수 있도록 도와준다.



그림 2.7.6 Scikit-learn Library

**3. 프로세스(기능) 설계**

**3.4 사용자 기능 설계(Sequence Diagram)**

**3.4.1 사용자 인증 – 회원가입**

사용자는 제공하는 모든 서비스를 이용하기 위해서는 회원가입을 통해 사용자 인증을 받아야한다. 회원가입 기능을 구현하기 위해 아래와 같이 설계한다.

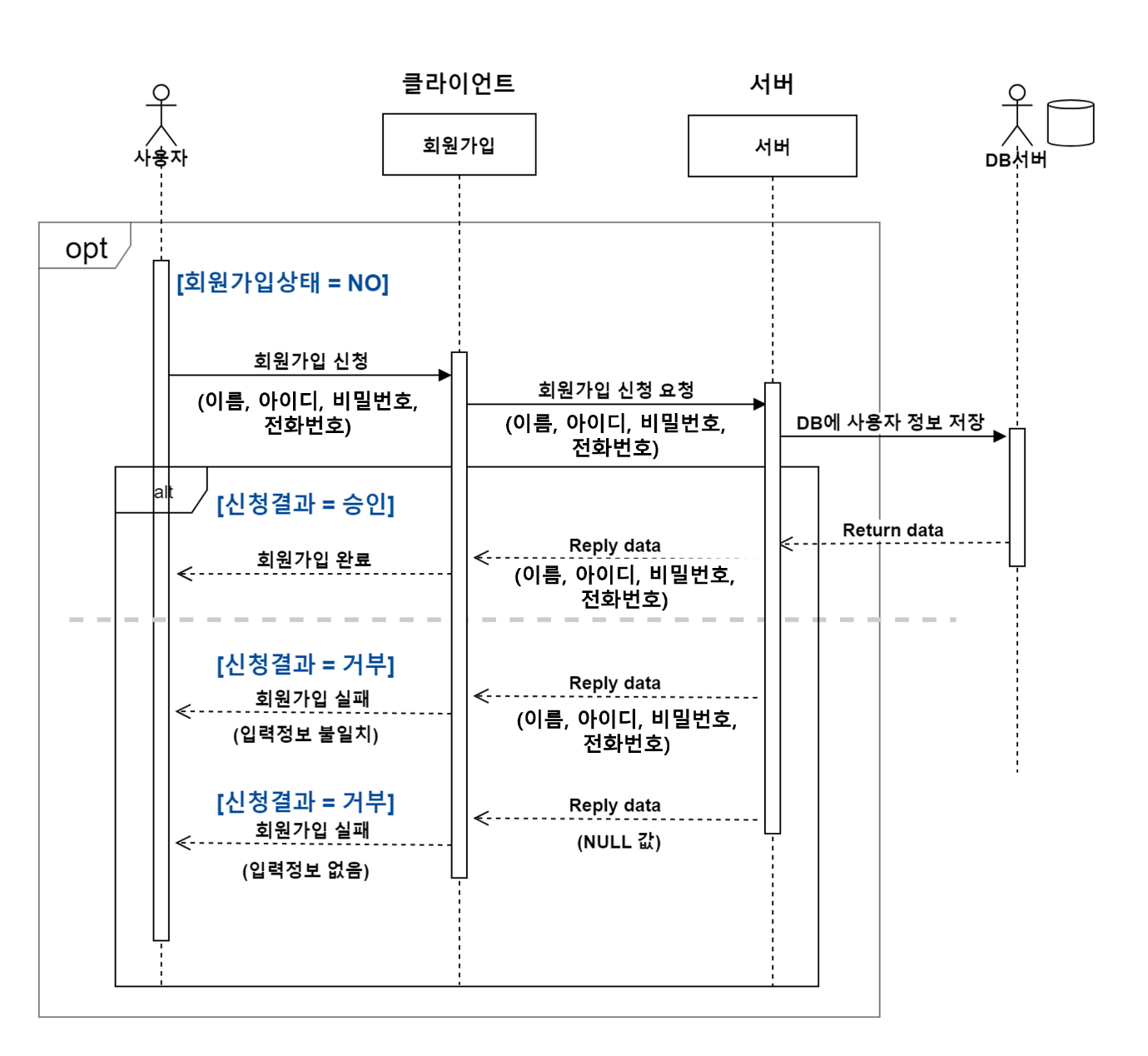


그림 3.4.1 사용자 인증 – 회원가입 Sequence Diagram

그림 3.4.1을 보면 사용자는 회원가입이 되어있지 않은 상태이다. 앱에서 회원가입 신청을 통해 서버에 회원가입 신청을 요청하고, 올바르게 정보를 입력하였다면 데이터베이스에 사용자 정보를 저장하고 회원가입을 승인받는다. 만약 사용자가 입력한 정보 중 비밀번호와 비밀번호 확인 값이 서로 다르면 승인을 거부하며, 입력정보가 없는 경우에도 승인을 거부한다. 이때, 사용자에게 실패 메시지와 함께 실패 사유를 나타낸다.

**3.4.2 사용자 인증 – 로그인**

사용자는 제공하는 모든 서비스를 이용하기 위해서는 로그인을 통해 사용자 인증을 받아야한다. 로그인 기능을 구현하기 위해 아래와 같이 설계한다.

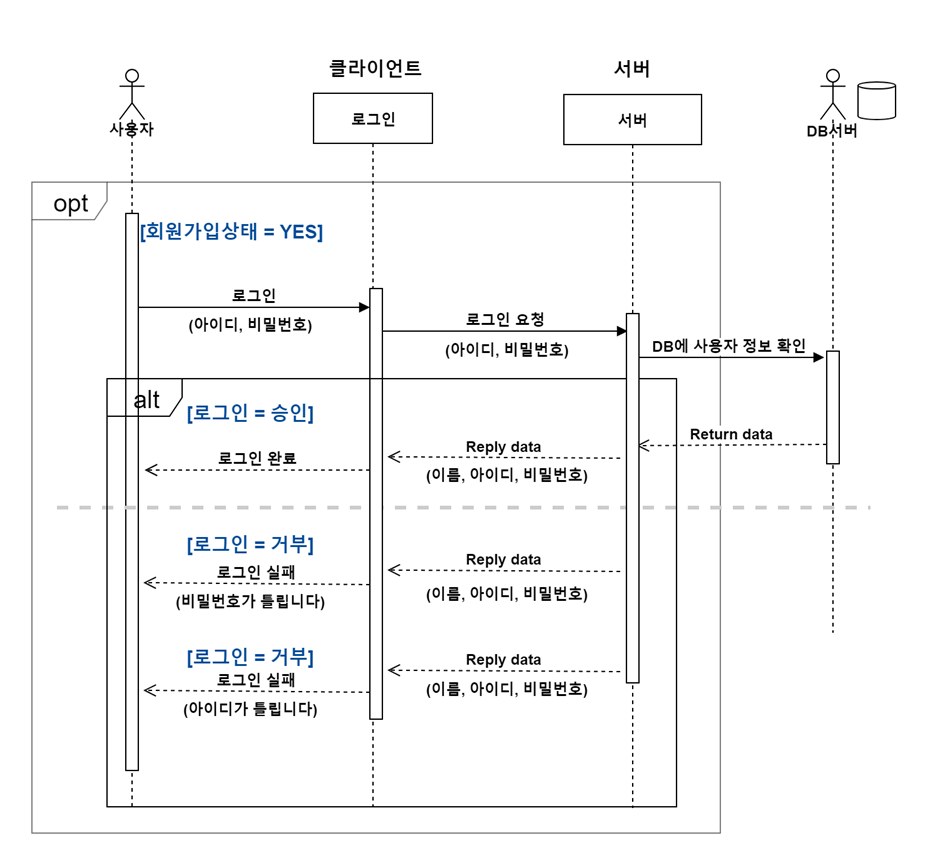


그림 3.4.2 사용자 인증 – 로그인 Sequence Diagram

그림 3.4.2을 보면 회원가입 상태가 되어있는 경우에 사용자는 아이디/비밀번호를 입력 후 로그인 요청을 한다. 앱에서는 사용자가 로그인 버튼을 클릭하면 서버에 로그인 요청을 한다. 입력받은 정보를 바탕으로 서버는 데이터베이스에 사용자 정보가 올바른지 요청하고, 요청받은 정보를 서버로 다시 되돌려준다. 아이디와 비밀번호가 일치한다면 로그인을 승인하고 사용자가 요청하는 서비스를 제공한다. 만약 로그인 요청 결과가 거부되었을 경우, 사용자에게 로그인 실패 메시지와 실패 사유를 함께 나타낸다.

**3.4.3 사용자 인증 – 로그아웃**

사용자는 로그인 후 서비스를 종료하기 위해서는 로그아웃 기능이 필요하다. 로그아웃 기능을 구현하기 위해 아래와 같이 설계한다.

텍스트, 도표, 스크린샷, 평행이(가) 표시된 사진

자동 생성된 설명

그림 3.4.3 사용자 인증 – 로그아웃 Sequence Diagram

그림 3.4.3을 보면 로그인이 되어있는 상태의 사용자는 서버에게 로그아웃 요청을 할 수 있다. 앱에서는 로그아웃 버튼을 클릭하거나 앱을 종료하면 서버에 로그아웃 요청을 한다. 로그아웃 요청을 받으면 서버는 사용자와의 연결을 끊는다. 로그아웃 버튼을 통해 로그아웃을 수행할 경우 로그아웃이 성공적으로 완료되었다는 내용과 함께 메시지를 제공한다.

**3.4.4 교통사고 신고 – 자동신고 온/오프**

사용자는 로그인 후 자동 신고를 활성화하거나 비활성화할 수 있다. 자동신고 온/오프 기능을 구현하기 위해 아래와 같이 설계한다.

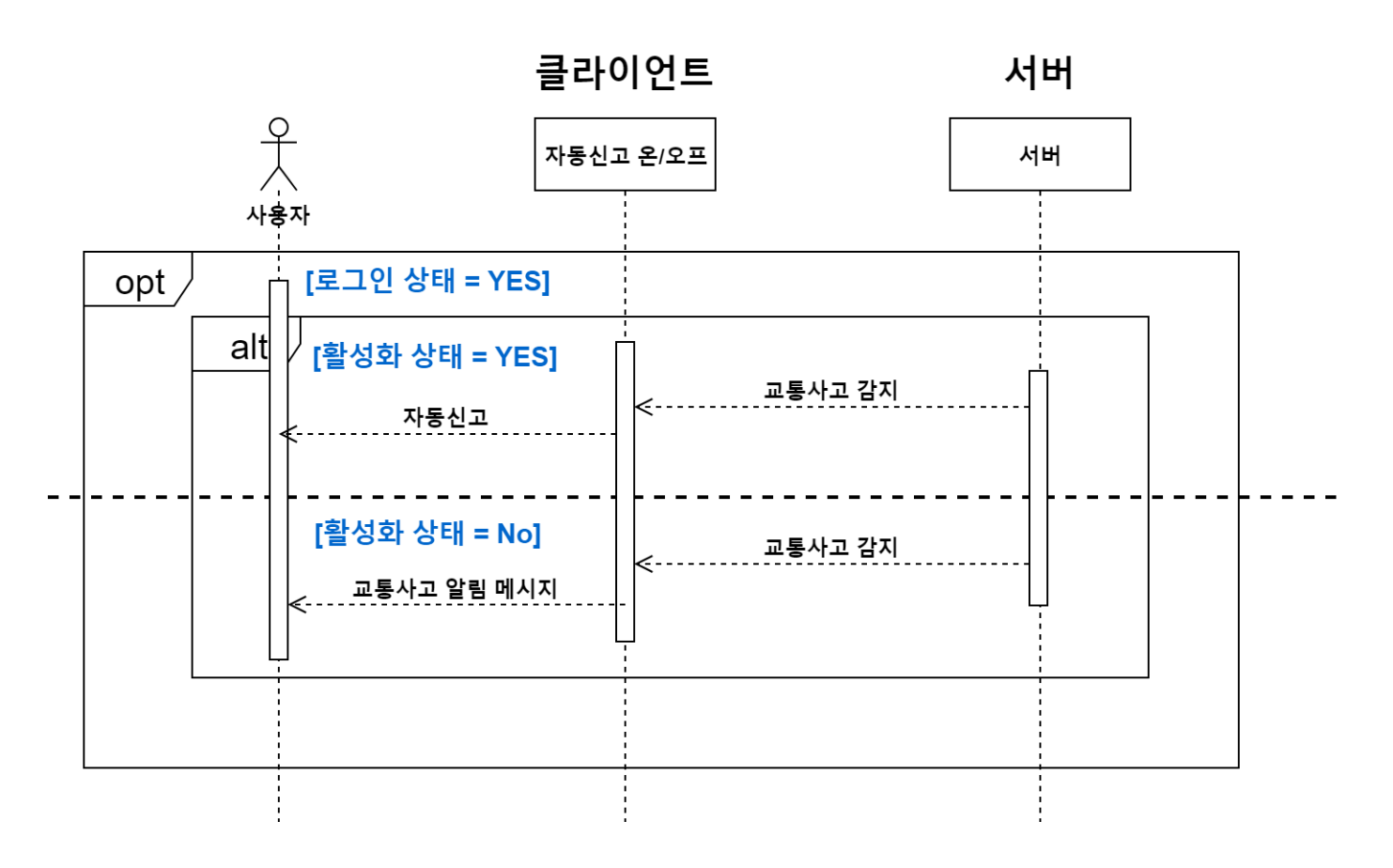


그림 3.4.4 교통사고 신고 – 자동신고 온/오프 Sequence Diagram

그림 3.4.4를 보면 로그인이 되어있는 상태의 사용자는 서버에게 자동신고 기능을 활성화하거나 비활성화 시킬 수 있다. 비활성화를 하면 서버는 교통사고가 감지될 경우 신고하지 않고, 알림 메시지를 나타낸다. 활성화를 하면 서버는 교통사고가 감지될 경우 자동으로 신고하고, 데이터베이스에 센싱 데이터를 저장한다.

**3.4.5 번호 관리 – 번호 등록**

사용자는 교통사고가 감지될 경우 사고 소식을 알릴 수 있는 번호를 등록할 수 있다. 번호 등록 기능을 구현하기 위해 아래와 같이 설계한다.

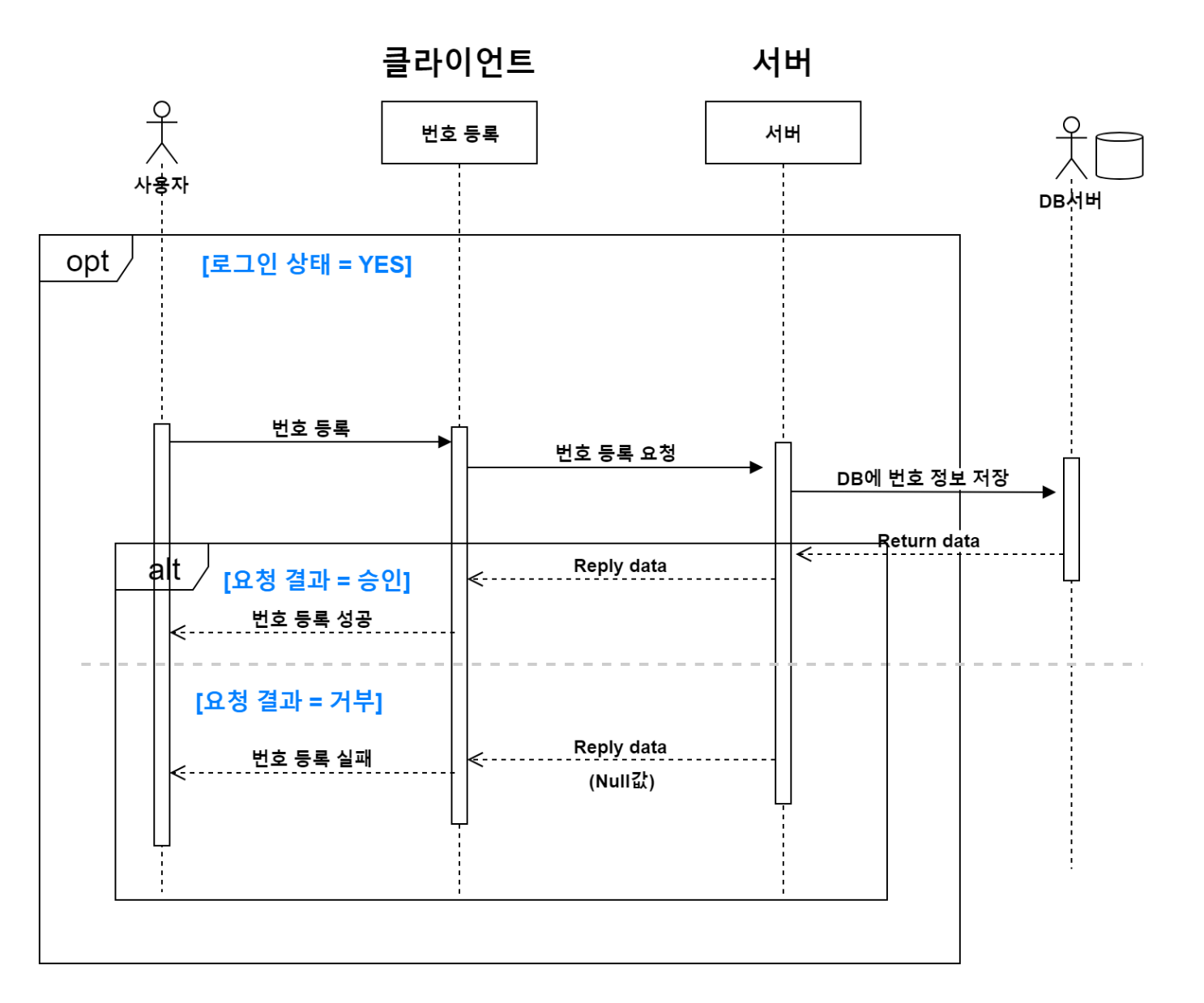


그림 3.4.5 번호 관리 – 번호 등록 Sequence Diagram

그림 3.4.5을 보면 로그인이 되어있는 상태의 사용자는 서버에게 번호 등록을 요청할 수 있다. 번호 등록을 요청받으면 서버는 사용자가 올바른 정보를 입력하였다면 데이터베이스에 번호 정보를 저장한다. 만약 사용자가 입력 정보가 잘못되었을 경우 승인을 거부하고, 사용자에게 실패 메시지와 실패 사유를 함께 나타낸다.

**3.4.6 번호 관리 – 번호 삭제**

사용자는 교통사고가 감지될 경우 사고 소식을 알릴 수 있는 번호를 삭제할 수 있다. 번호 삭제 기능을 구현하기 위해 아래와 같이 설계한다.

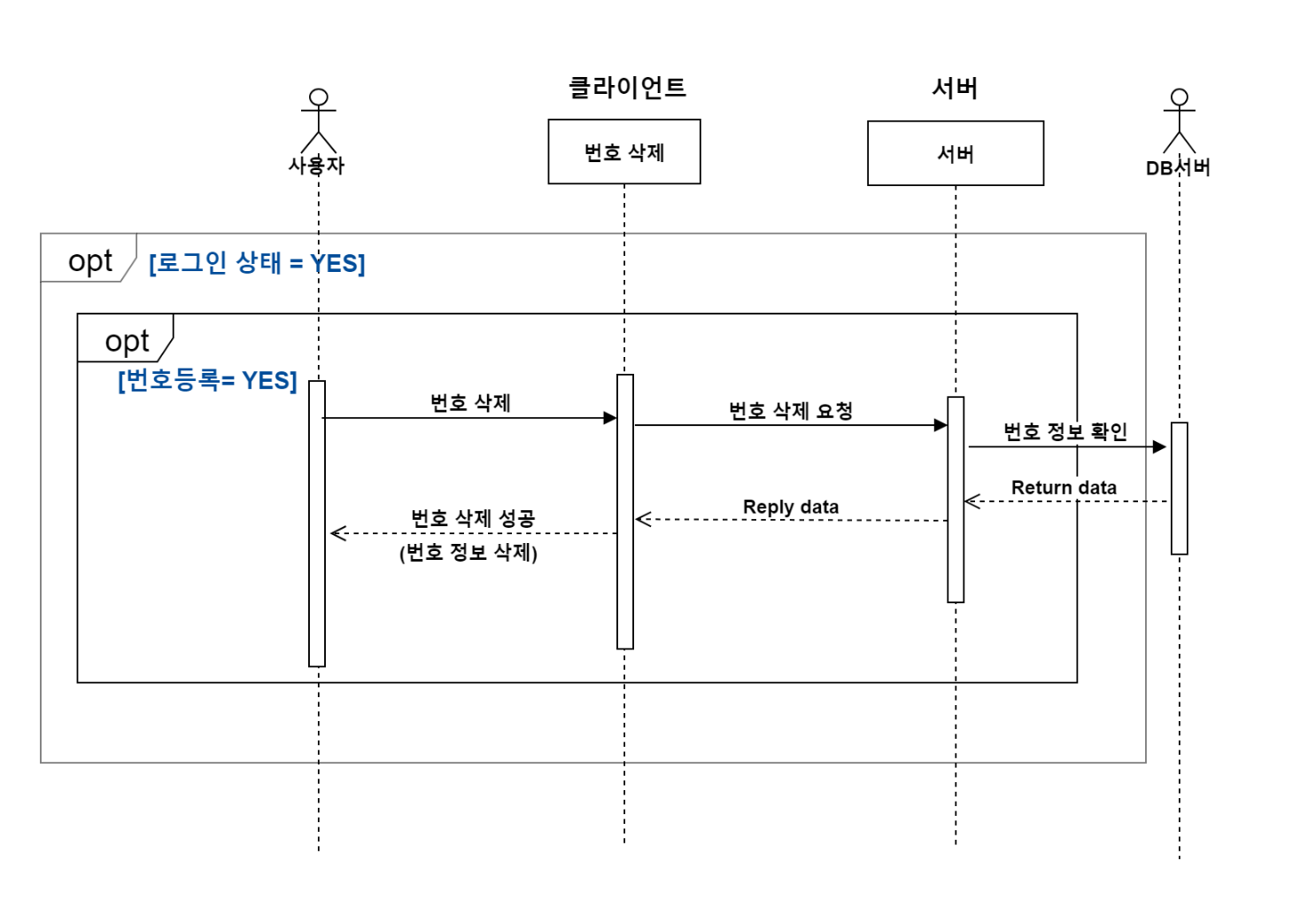


그림 3.4.6 번호 관리 – 번호 삭제 Sequence Diagram

그림 3.4.6을 보면 로그인이 되어있고, 번호를 등록한 상태의 사용자는 서버에게 번호 삭제 요청을 할 수 있다. 서버는 상품 삭제 요청을 받으면 데이터베이스에 번호 정보를 확인하여 번호 삭제가 성공적으로 완료되었다는 내용과 함께 메시지를 제공한다.

**3.5 관리자 기능 설계(Sequence Diagram)**

**3.5.1 관리자 인증 – 로그인**

관리자는 서비스를 관리하기 위해 본인의 아이디/비밀번호를 이용한 로그인이 필요하다. 로그인 기능을 구현하기 위해 아래와 같이 설계한다.

텍스트, 도표, 폰트, 스크린샷이(가) 표시된 사진

자동 생성된 설명

그림3.5.1 관리자 인증 – 로그인 Sequence Diagram

그림 3.5.1을 보면 회원가입 상태가 되어있는 경우에 관리자는 아이디/비밀번호를 입력 후 로그인 요청을 한다. 앱에서는 관리자가 로그인 버튼을 클릭하면 서버에 로그인 요청을 한다. 입력받은 정보를 바탕으로 서버는 데이터베이스에 사용자 정보가 올바른지 요청하고, 요청받은 정보를 서버로 다시 되돌려준다. 아이디와 비밀번호가 일치한다면 로그인을 승인하고 관리자에게 관리자 홈을 제공한다. 만약 로그인 요청 결과가 거부되었을 경우, 관리자에게 로그인 실패 메시지와 실패 사유를 함께 나타낸다.

**3.5.2 관리자 인증 – 로그아웃**

관리자는 로그인 후 서비스를 종료학 위해서는 로그아웃 기능이 필요하다. 로그아웃 기능을 구현하기 위해 아래와 같이 설계한다.

텍스트, 도표, 스크린샷, 번호이(가) 표시된 사진

자동 생성된 설명

그림 3.5.2 관리자 인증 – 로그아웃 Sequence Diagram

그림 3.5.2를 보면 로그인이 되어있는 상태의 관리자는 서버에게 로그아웃 요청을 할 수 있다. 앱에서는 로그아웃 버튼을 클릭하거나 앱을 종료하면 서버에 로그아웃 요청을 한다. 로그아웃 요청을 받으면 서버는 관리자와의 연결을 끊는다. 로그아웃 버튼을 통해 로그아웃을 수행할 경우 로그아웃이 성공적으로 완료되었다는 내용과 함께 메시지를 제공한다.

**3.5.3 회원 관리 – 회원 정보 조회**

관리자는 회원 관리를 위해 회원 정보를 조회할 수 있는 권한을 가지고 있다. 회원 정보 조회 기능을 구현하기 위해 아래와 같이 설계한다.

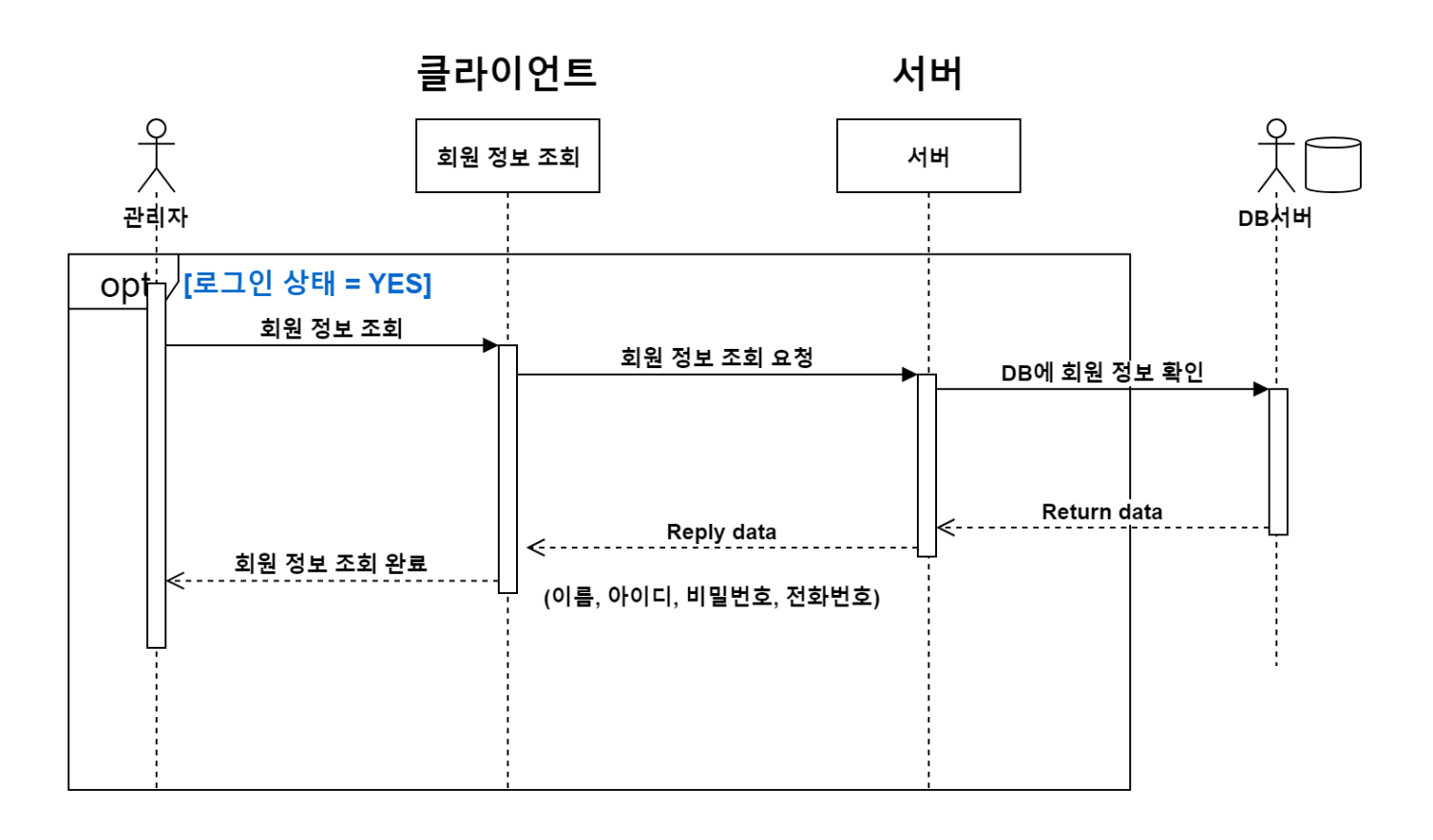


그림 3.5.3 회원 관리 – 회원 정보 조회 Sequence Diagram

그림 3.5.3을 보면 로그인이 되어있는 상태의 관리자는 서버에게 회원 정보 조회 요청을 할 수 있다. 서버는 회원 정보 조회 요청을 받으면 데이터베이스에 저장된 회원 정보를 확인하여 회원 정보 데이터를 테이블 형태로 관리자에게 보여준다.

**3.5.4 센싱 데이터 관리 – 센싱 데이터 조회**

관리자는 센싱 데이터 관리를 위해 센싱 데이터를 조회할 수 있는 권한을 가지고 있다. 센싱 데이터 조회 기능을 구현하기 위해 아래와 같이 설계한다.

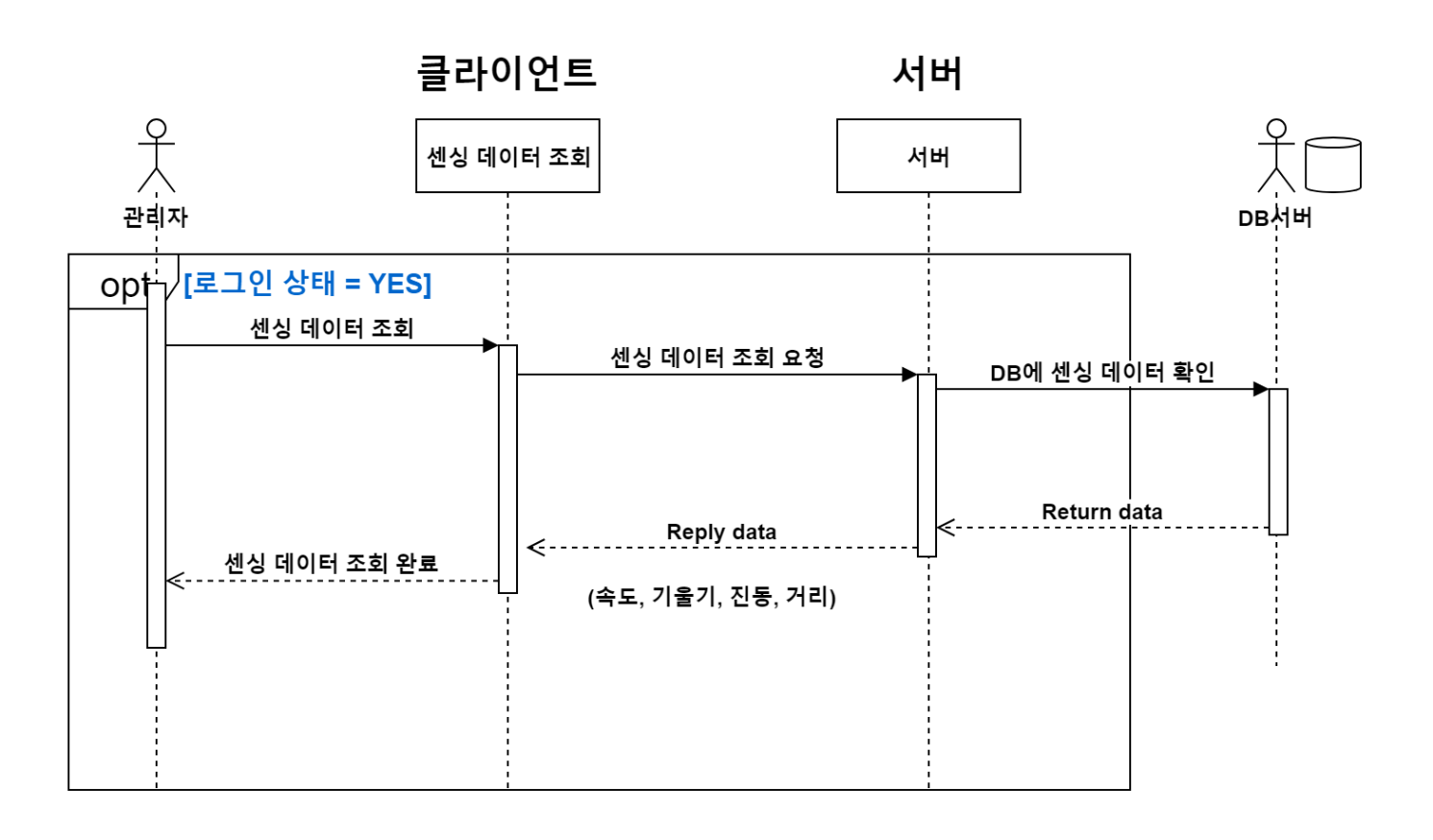


그림 3.5.4 센싱 데이터 관리 – 센싱 데이터 조회 Sequence Diagram

그림 3.5.4를 보면 로그인이 되어있는 상태의 관리자는 서버에게 센싱 데이터 조회 요청을 할 수 있다. 서버는 센싱 데이터 조회 요청을 받으면 데이터베이스에 저장된 센싱 데이터를 확인하여 센싱 데이터를 테이블 형태로 관리자에게 보여준다.

**3.6 AI server 기능 설계(Sequence Diagram)**

**3.6.1 교통사고 분류 기능**

사용자는 교통사고가 감지되었을 경우 교통사고 알림 기능을 제공받을 수 있다. 교통사고 분류 기능을 구현하기 위해 아래와 같이 설계한다.

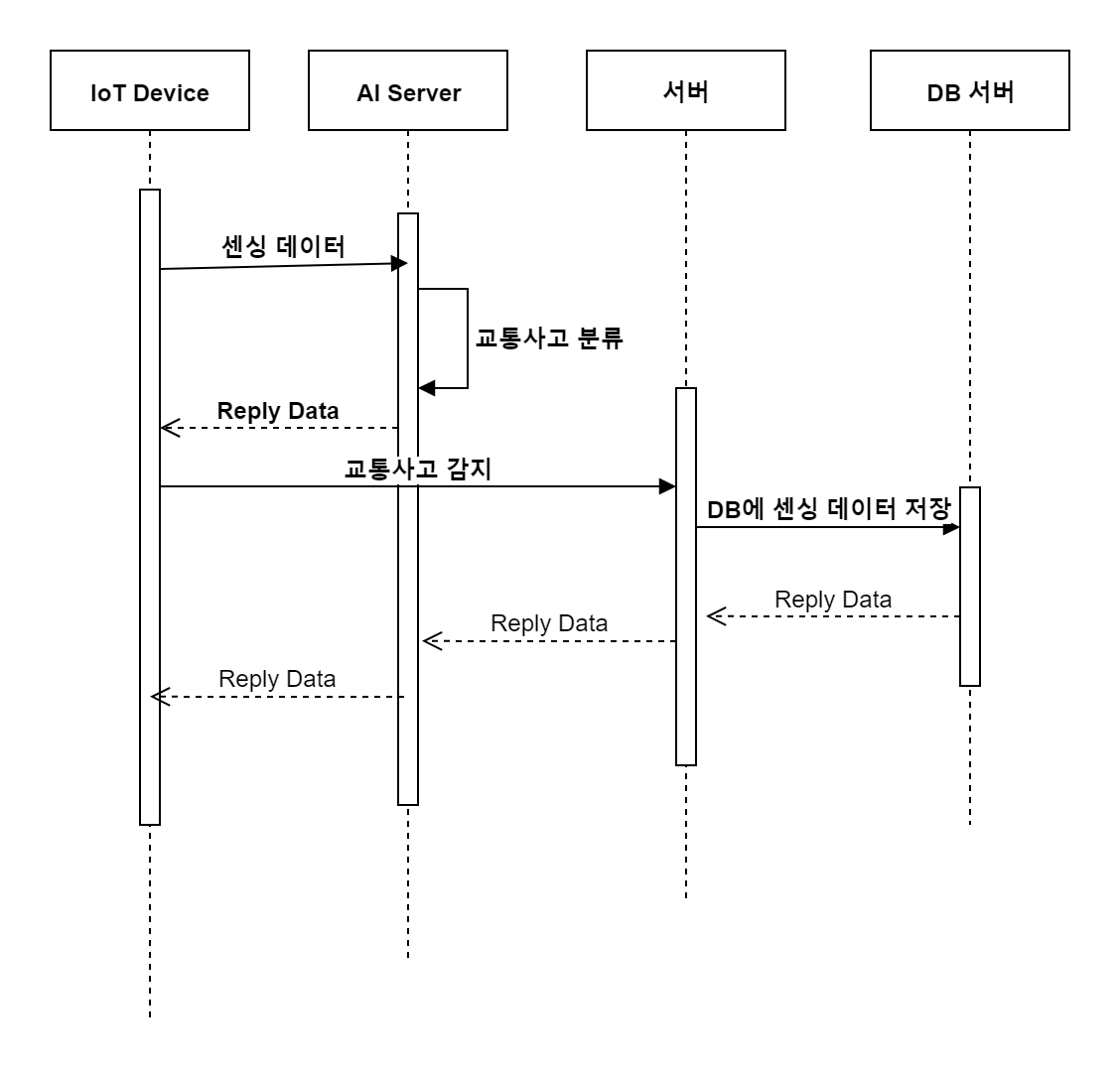


그림 3.6.1 교통사고 분류 기능 Sequence Diagram

그림 3.6.1을 보면 IoT Device는 Tick Rate마다 데이터를 측정한다. 그 데이터는 AI Server로 전송되고, AI Server는 자체 알고리즘을 통해 교통사고를 판별한다. 교통사고로 판별될 경우 서버에 알리고, 서버는 데이터베이스에 센싱 데이터를 저장한다.

**7. 환경구성**

**7.1 개발환경 및 운영환경**

**● 개발환경(IoT)**

|  |  |  |  |
| --- | --- | --- | --- |
| 구분 | 개발환경 | 규격 | 비고 |
| 개발툴 | 편집툴 | ATOM |  |
| 언어 | Python |
| 개발IDE | Pycharm, VSC |
| 프레임워크 | Anaconda |
| 사용할 미들웨어(모듈) | Pandas  Matplotlib  NumPy  Jupyter |
| 서버 | OS | Rasphian |
| DB | Firebase |
| 서버 하드웨어 | CPU | ARM |  |
| RAM | 1GB SRAM |
| DISK | MicroSD |
| Network | 100Mbps |

**● 개발환경(서버 또는 모바일)**

|  |  |  |  |
| --- | --- | --- | --- |
| 구분 | 개발환경 | 규격 | 비고 |
| 개발툴 | 편집툴 | ATOM |  |
| 언어 | Java |
| 개발IDE | Android Studio |
| 프레임워크 | Android Framework |
| 사용할 미들웨어(모듈) | GSON |
| 서버 | OS | Windows |
| DB | Firebase |
| 서버 하드웨어 | CPU | Intel Core i7 |  |
| RAM | 8 RAM |
| DISK | 512GB |
| GPU | Geforce RTX 2060 super |

**● 운영환경**

|  |  |  |  |
| --- | --- | --- | --- |
| 구분 | 개발환경 | 규격 | 비고 |
| 클라이언트 | OS | Android |  |
| 하드웨어 | CPU | Intel Core i7 |  |
| RAM | 8 RAM |
| DISK | 512GB |
| Network | 100Mbps |

**7.2 소스디렉터리 구조**

**● 개발환경(IoT)**

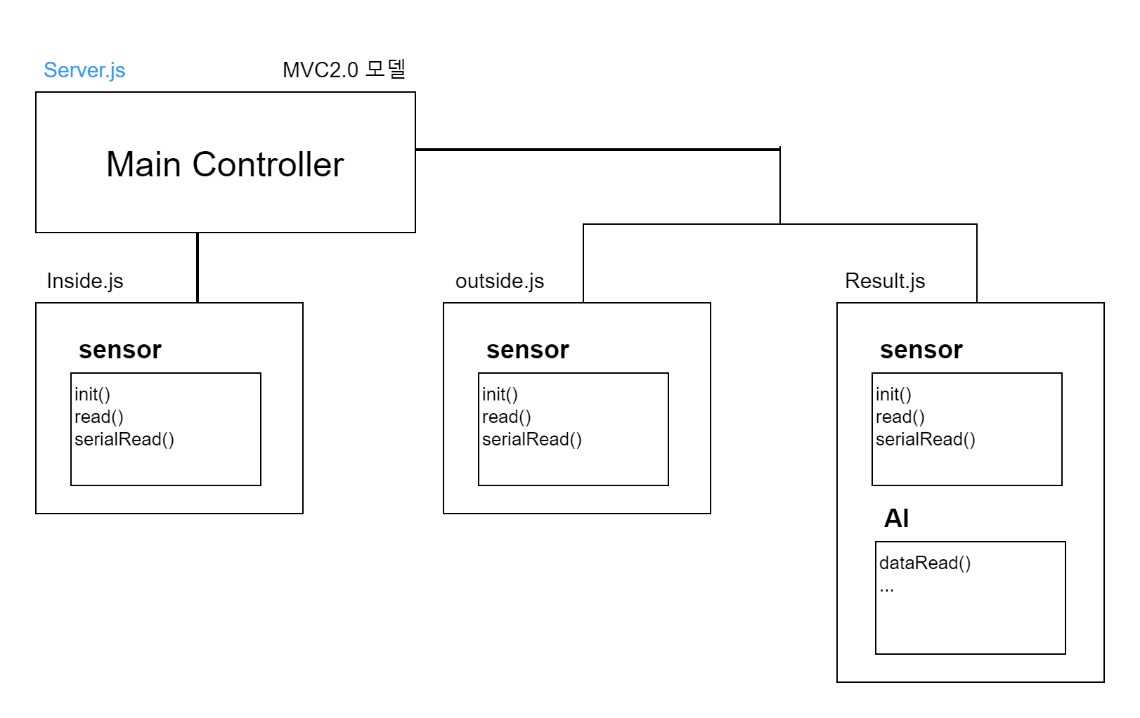


그림 7.2.1 MVC2 모델 구조

구현에 앞서, 그림 7.2.1와 같이 MVC 구조로 개발 디렉터리 구조를 정의한다.

텍스트, 폰트, 라인, 도표이(가) 표시된 사진

자동 생성된 설명

그림 7.2.2 디렉터리 및 파일 환경

그림 7.2.2와 같이 세부 디렉터리와 파일명으로 모듈화를 정의한다.

**● 개발환경(서버 또는 모바일)**

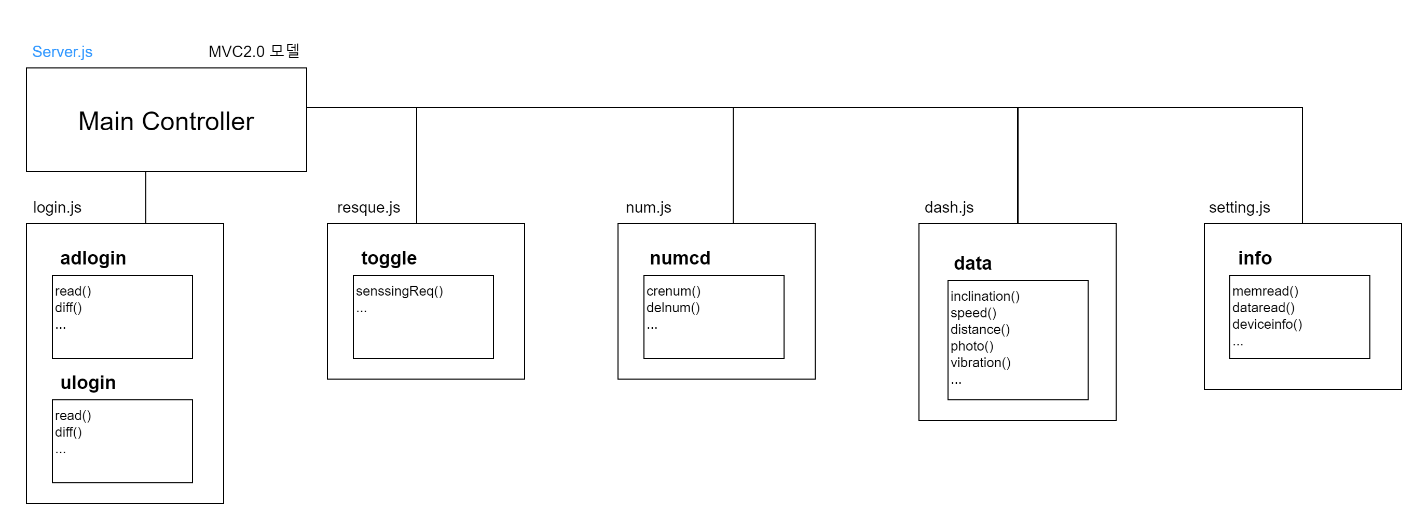


그림 7.2.3 MVC2 모델 구조

구현에 앞서, 그림 7.2.3와 같이 MVC 구조로 개발 디렉터리 구조를 정의한다.

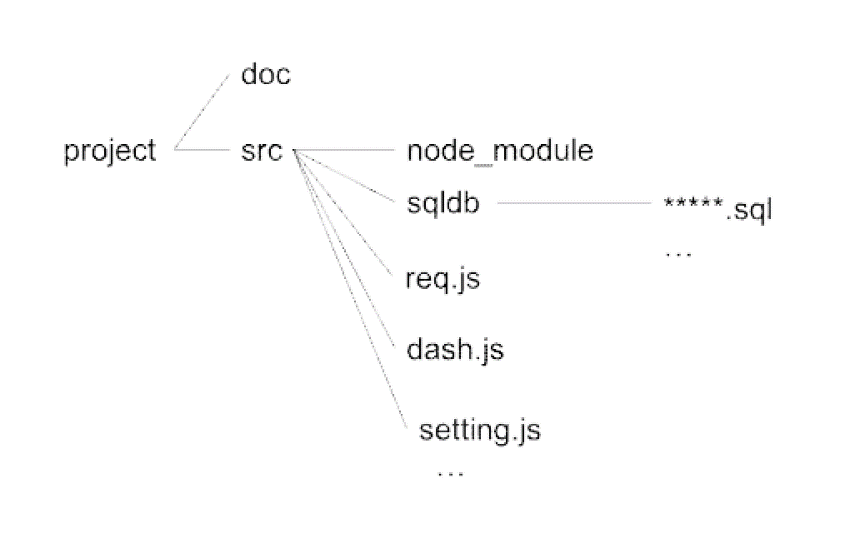


그림 7.2.4 디렉터리 및 파일 환경

그림 7.2.4와 같이 세부 디렉터리와 파일명으로 모듈화를 정의한다.